OBJECT ORIENTED TECHNOLOGY in AVIATION

Object-Oriented Software

Object-Oriented Technology in Aviation (OOTiA) Survey

Software Conference
Norfolk, Virginia

July 27, 2005

OOTiA Survey

Background

- The FAA kicked off an effort to develop internal training for ACO engineers, other FAA technical folks (and eventually SW DERs) involved with software approvals

- The training effort kick-off coincided with the release of the OOTiA Handbook and the training was to address both OO techniques generally as well as Handbook issues

- As the training developed, more and more applicants were becoming aware of the Handbook and its potential strengths and liabilities

- To provide a *qualitative* sense of perspective, a survey was developed and distributed (via the FAA’s e-mail database) to determine where the industry might be in terms of maturity and familiarity with the Handbook

OOTiA Survey

Nature of the Survey

- The survey had twenty-one questions intended to solicit information in the following major areas via short answers:
  - Are you using OO techniques today and if so, at what software level and what language(s)?
  - If you are utilizing inheritance and dynamic dispatch, how are you dealing with verification issues?
  - How are things going and have the certification authorities been involved?
  - What would you like to see in terms of tools developed by third-party suppliers?
  - How has (or has not) the Handbook been useful?
  - What improvements could be made in terms of the Handbook or other guidance material?

OOTiA Survey

Nature of the Survey Responses

- There were twenty-six respondents with three respondents deferring on providing input – respondents with inputs included the following:
  - Aircraft OEMs (all major OEMs in both Europe and US represented)
  - Avionics suppliers (all major US and Canadian suppliers represented)
  - Consulting DERs
  - Third-party development and verification organizations
- In some cases, there were multiple responses from different organizations of the same corporation – ironically, the responses were very different across each organizational entity of the same parent corporation
- Reponses ranged from “evidently well informed” to “I almost know how to spell OO” with a number of responses somewhere in-between

OOTiA Survey

Survey and Presentation Limitations

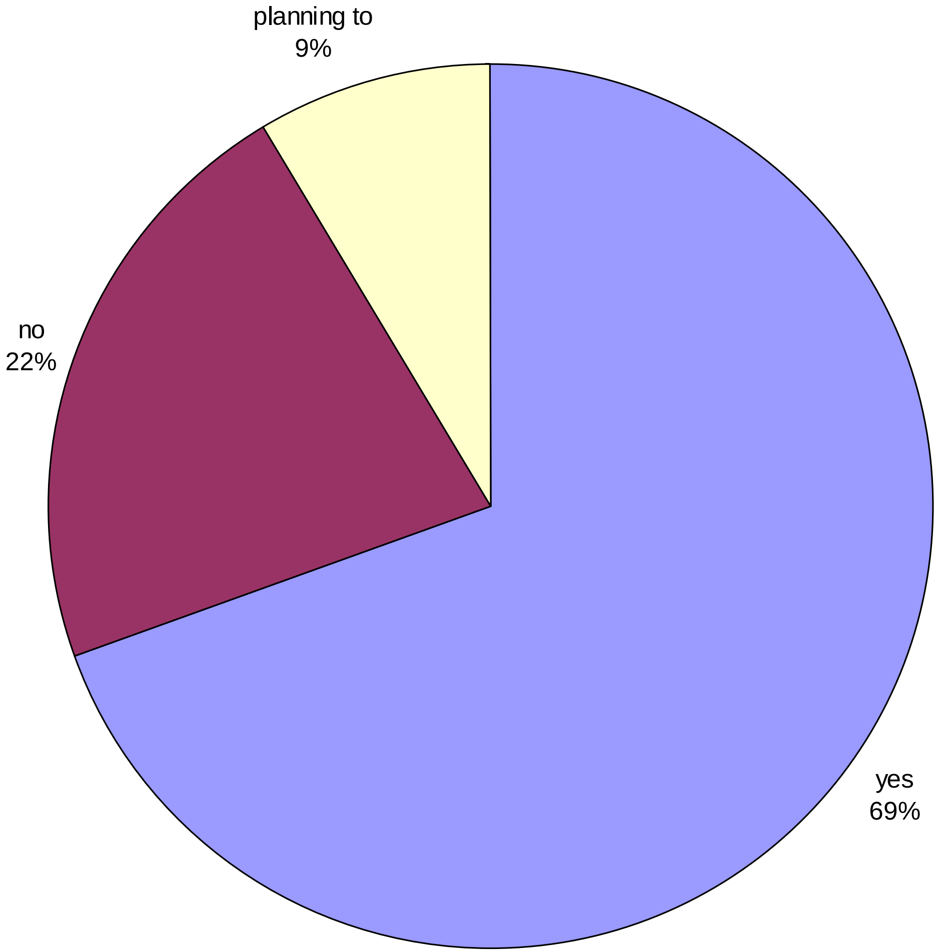
- The survey cannot be considered a scientific poll – no true sampling was achieved
- Respondents did not always appear to be fully cognizant of some OO terminology that was used (Did the right people complete the survey?)
- The results compiled for this presentation were quantized whereas actual responses were typically in the form of short answers – there could be some variability in how the quantization was performed

"Raw" survey results, sanitized for company names and so on, will be placed on the SC-205 website

OOTiA Survey

Survey Responses

Are you using OO languages in your programs?

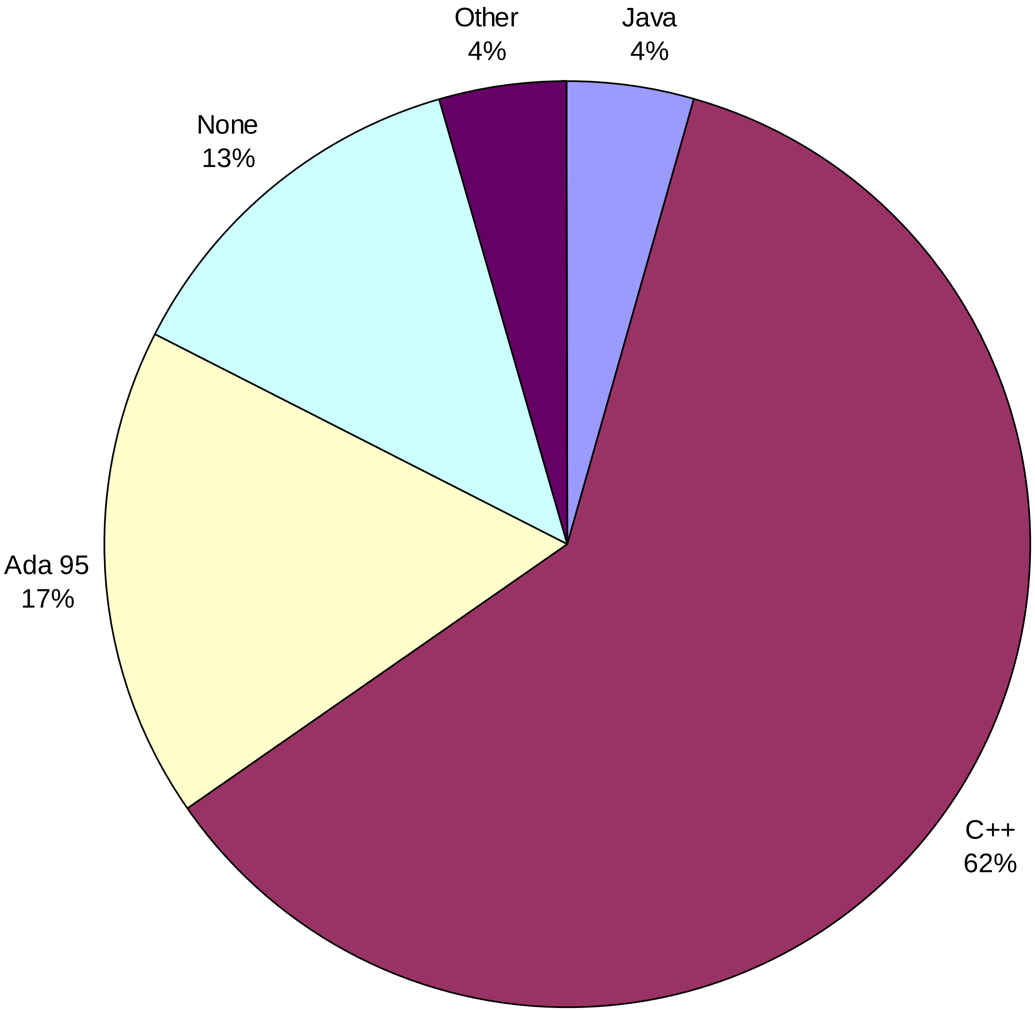


OOTiA Survey

Survey Responses

What OO language do you use predominately?

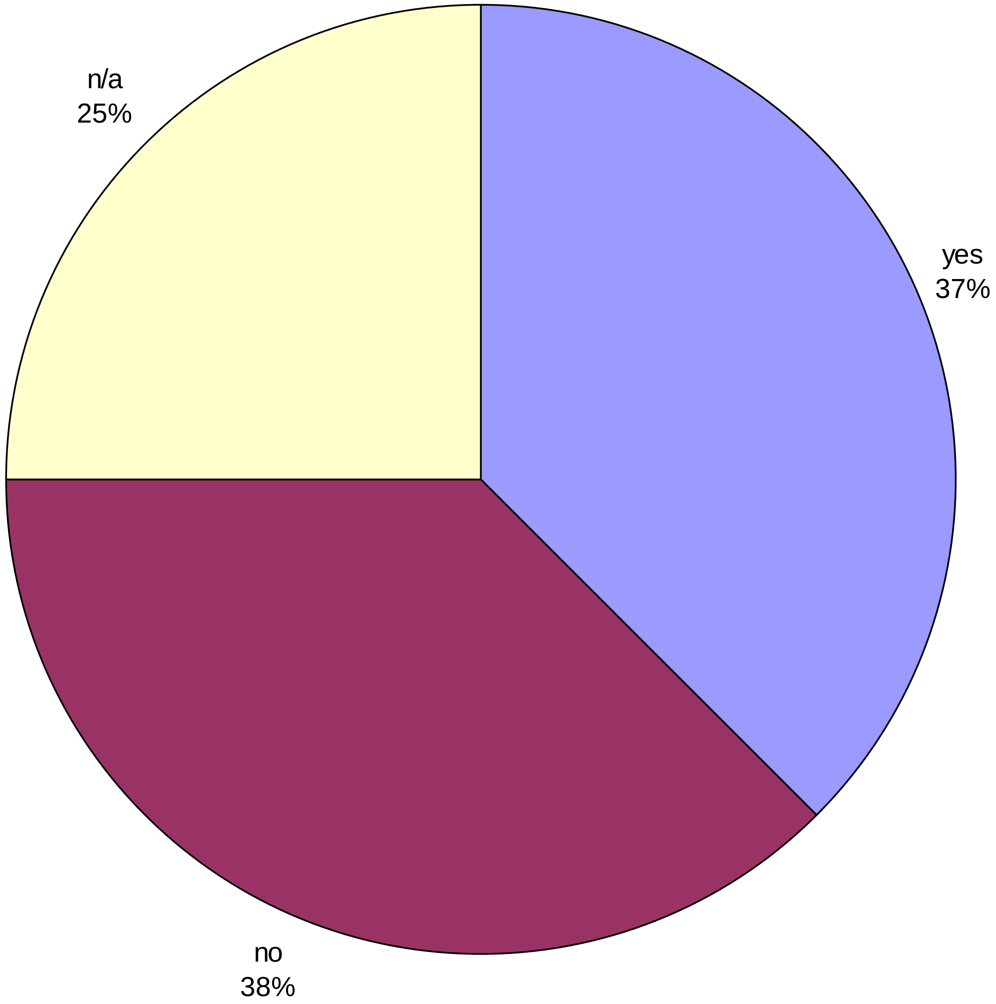
(Note: Several respondents utilized both Ada and C++)



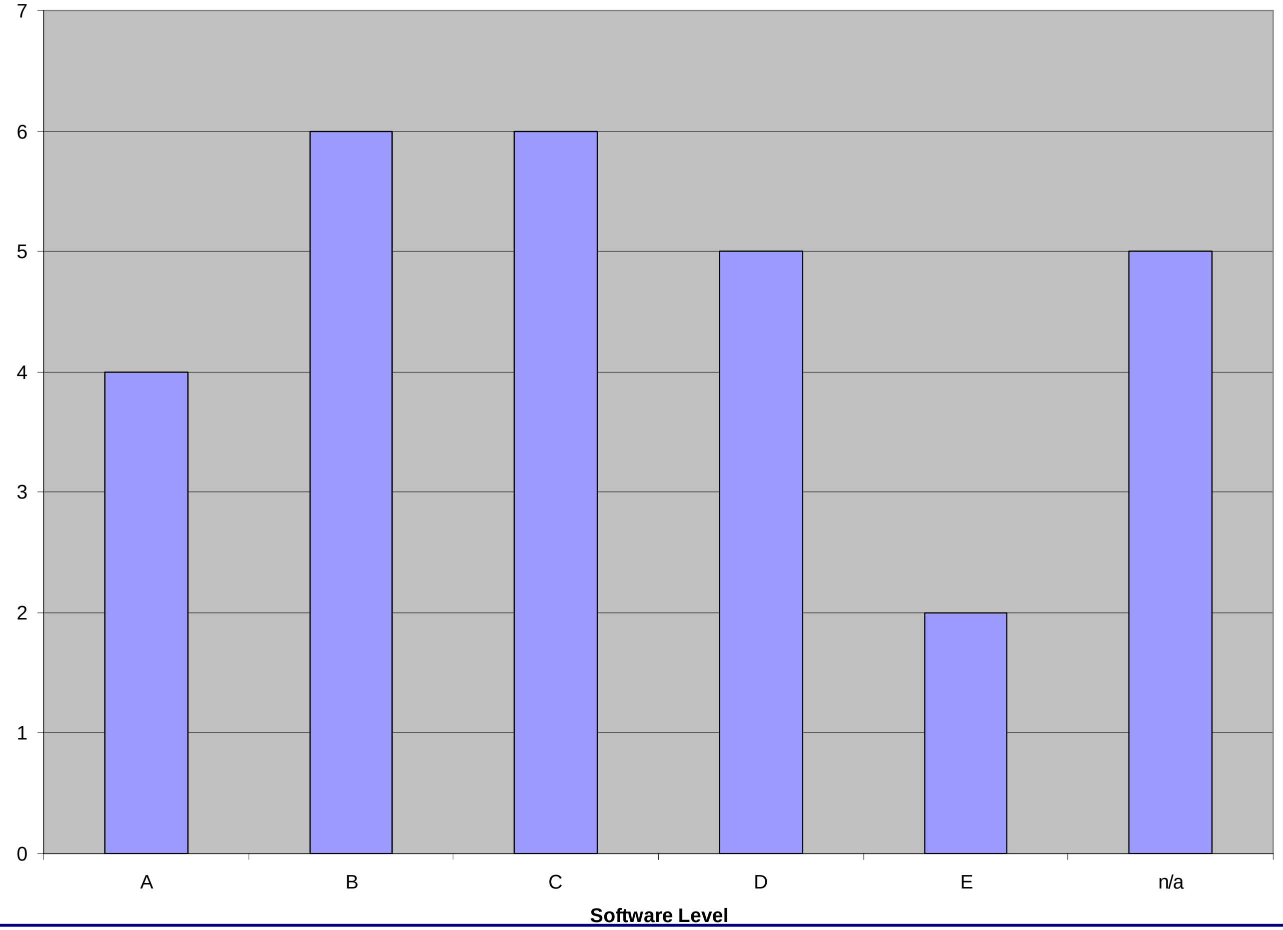
OOTiA Survey

Survey Responses

Are any of the applications being developed using OO at Level B or above?



What is the criticality level (A, B, C, D, E) of the majority of your applications that are utilizing OO techniques?

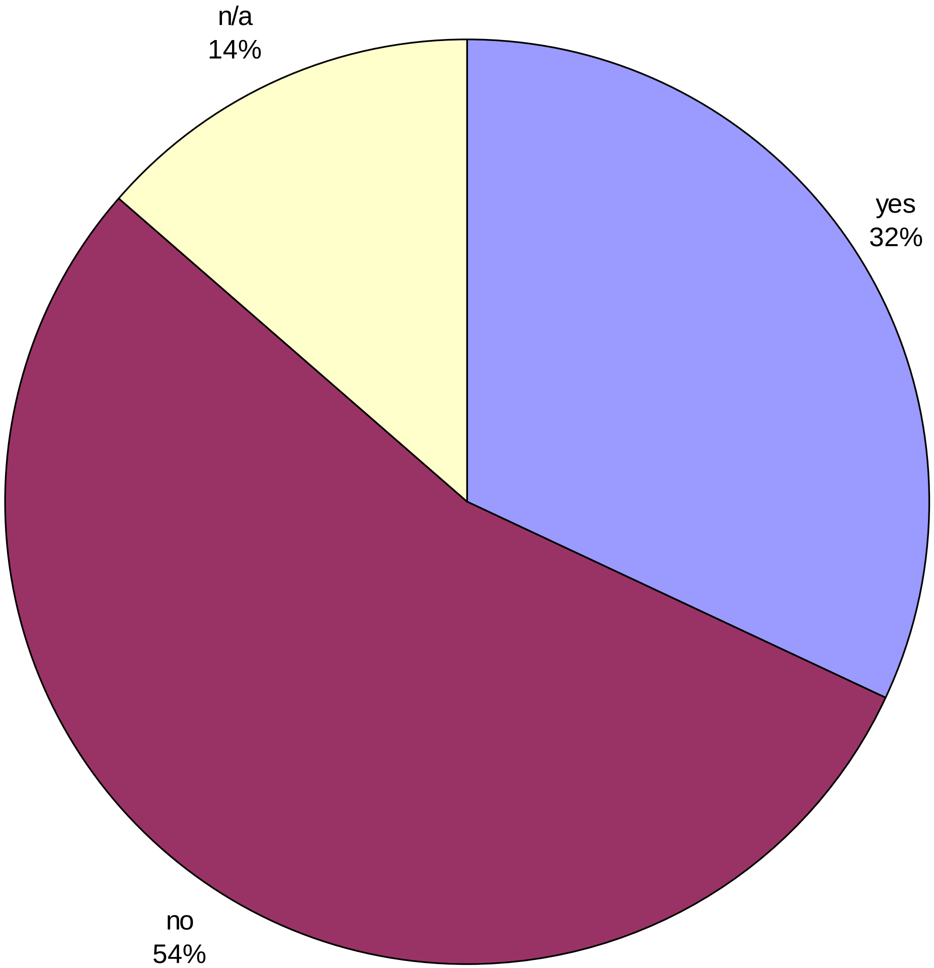


(Note: Some respondents are working with multiple software levels)

OOTiA Survey

Survey Responses

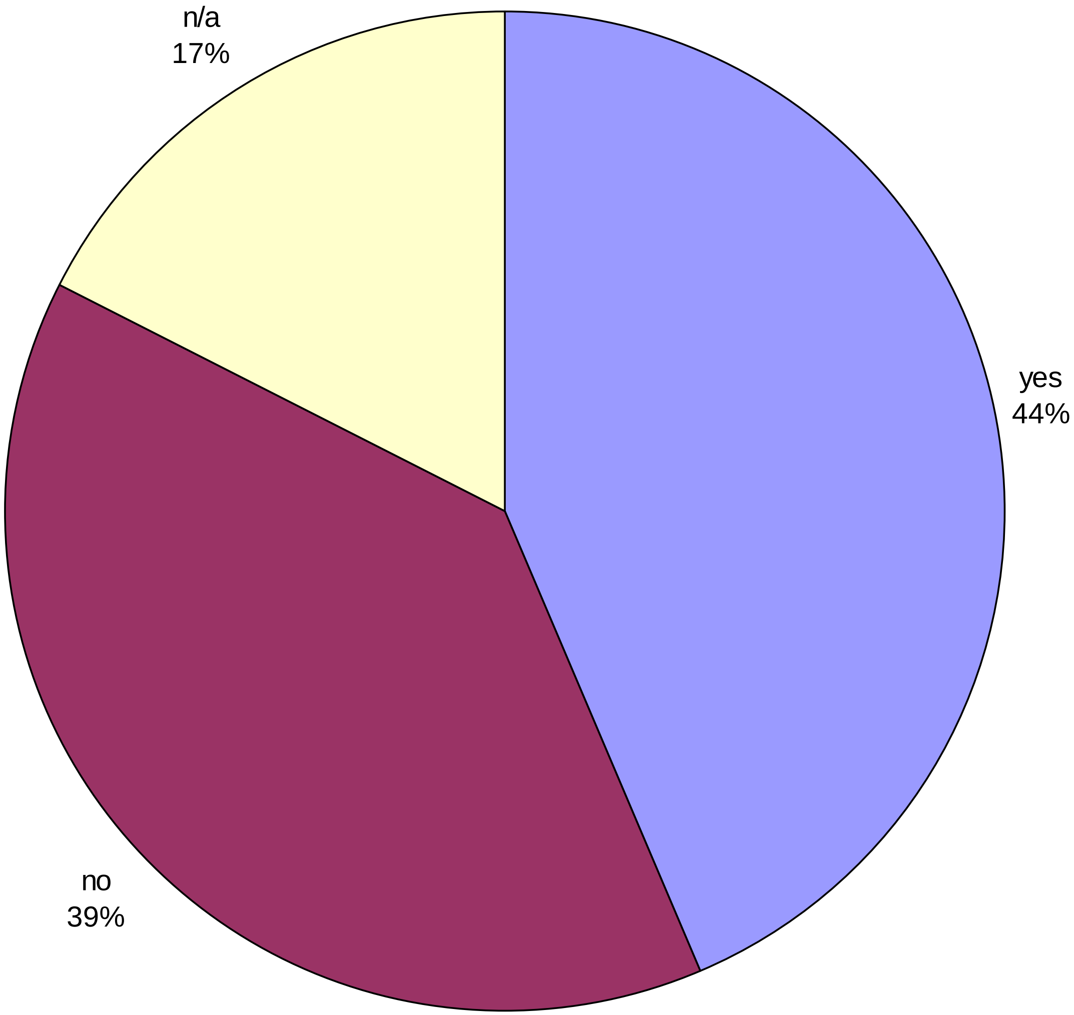
Have you had an issue paper or CRI issued against the OO effort?



OOTiA Survey

Survey Responses

Have you had active certification authority involvement in your program or have the certification authorities performed a review on the OO aspects of your program?



OOTiA Survey

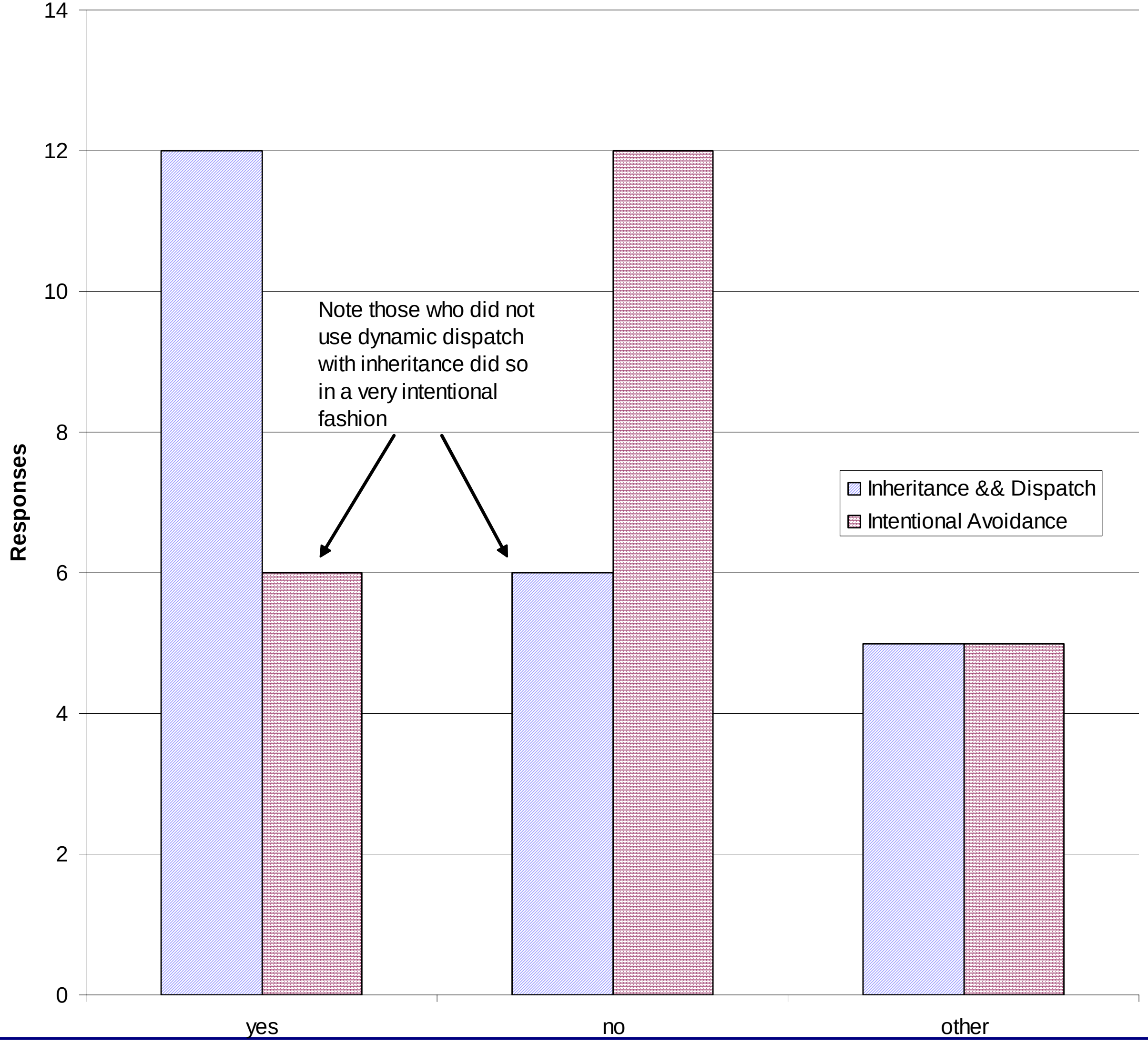
Survey Responses

Are you utilizing inheritance and dynamic dispatch in your implementation?

If not, did you avoid inheritance and dynamic dispatch intentionally?

If you chose to avoid inheritance and dynamic dispatch, if possible, please provide your rationale.

Most frequent rationale provided was to simplify verification and/or lower Cert Authority concerns

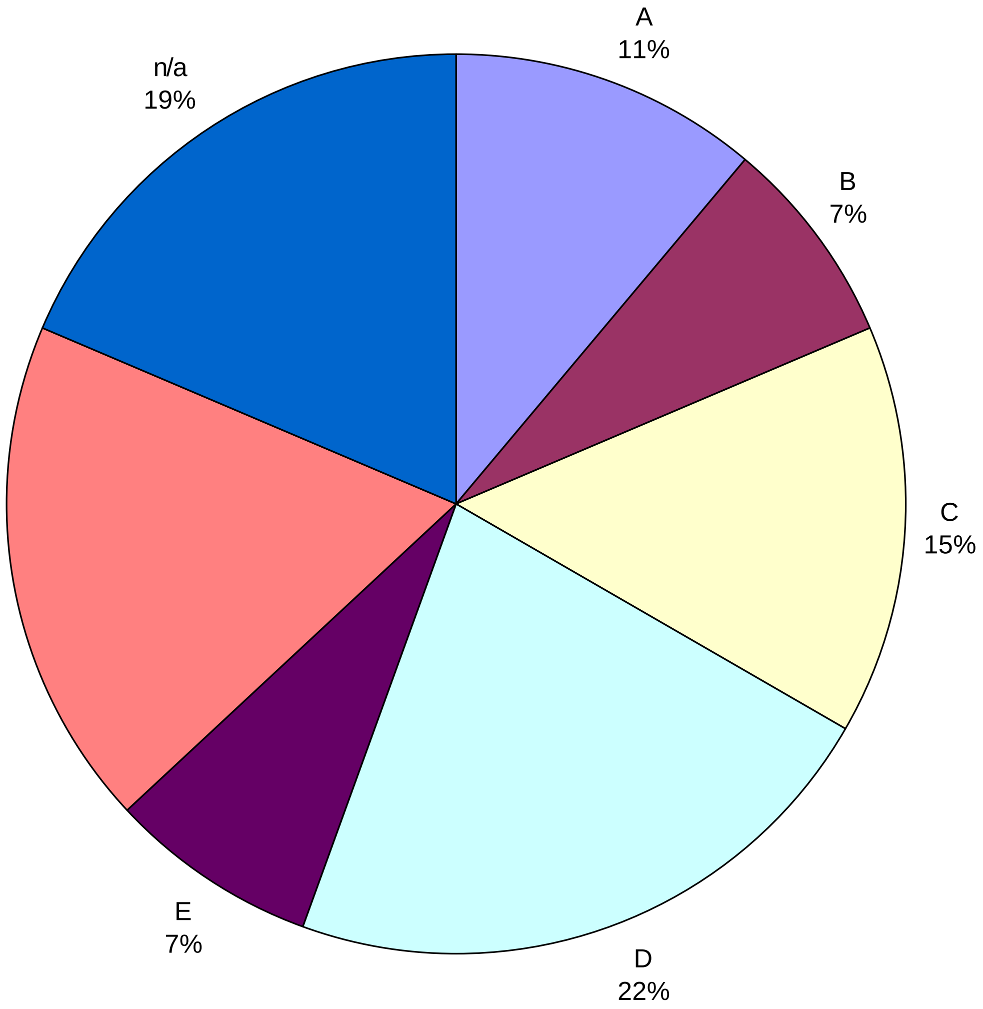


Note those who did not use dynamic dispatch with inheritance did so in a very intentional fashion

OOTiA Survey

Survey Responses

What software levels are you developing that use inheritance and dynamic dispatch (A,B,C,D)?

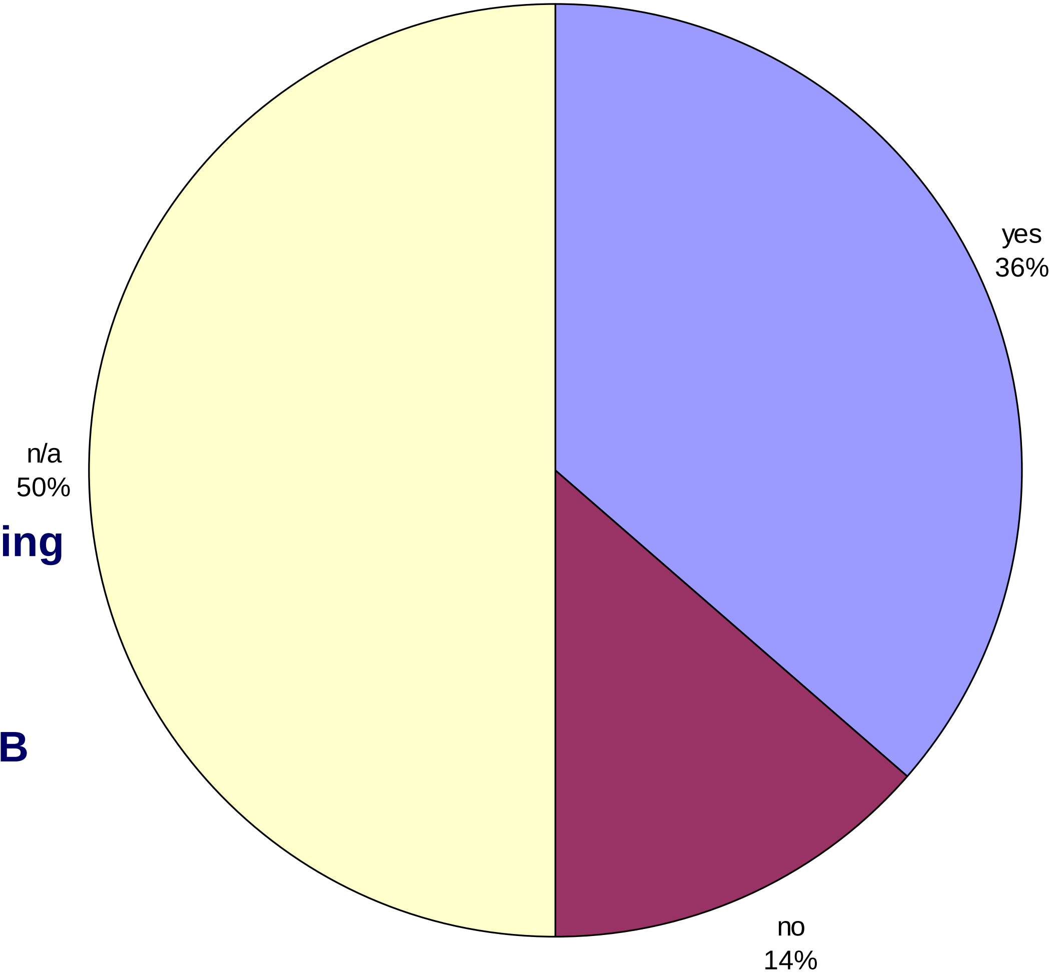


Note: There were some minor inconsistencies across responses in the general area of inheritance and dynamic dispatch

OOTiA Survey

Survey Responses

If you are using inheritance and dynamic dispatch, did you consider the information presented in Volume 3 of the OOTiA Handbook for inheritance and dynamic dispatch (substitutability guidelines)?



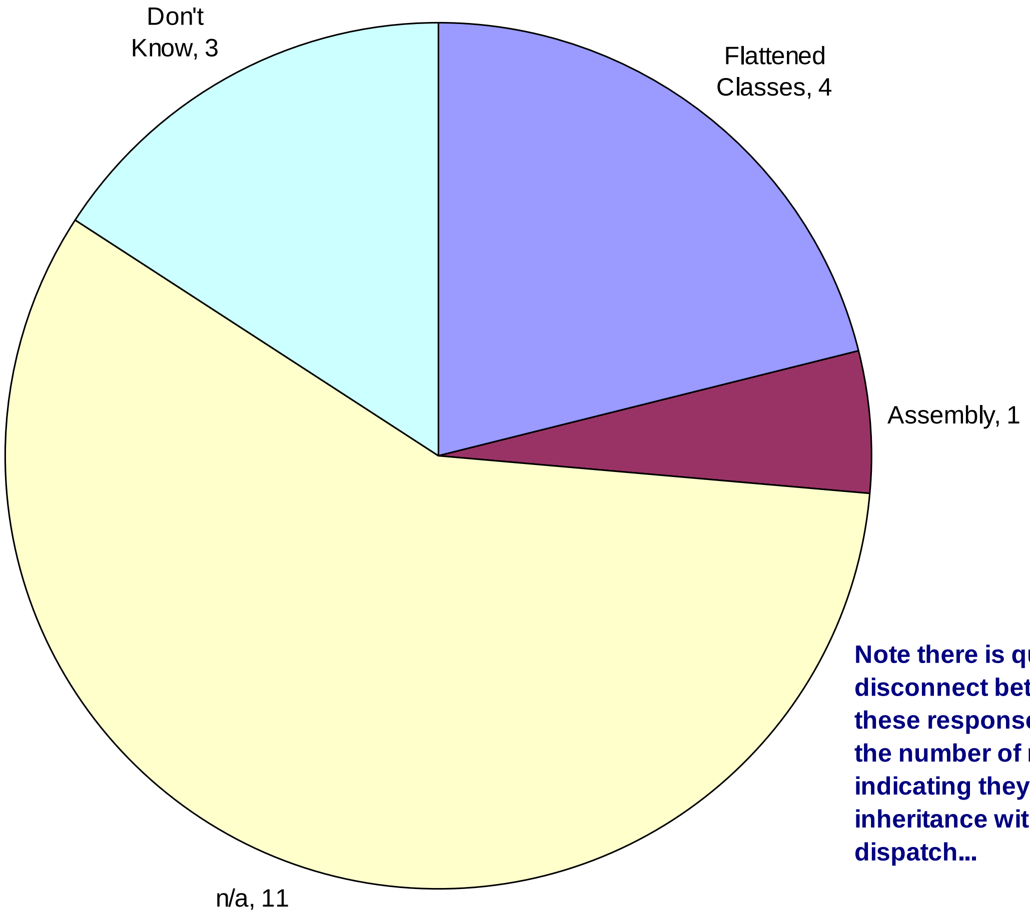
For the "no" responses, the following rationale was provided:

- **OOTiA provided no guidance for meeting the objectives of DO-178B**
- **Made use of FAA white papers**
- **Level D; n/a**

OOTiA Survey

Survey Responses

If you are using inheritance and dynamic dispatch for levels A, B, or C, are you using a “flattened class” approach for testing and structural coverage analysis, or are you pursuing another means of verification such as structural coverage at the assembly language level combined with demonstrating dispatch tables are fully covered via test?

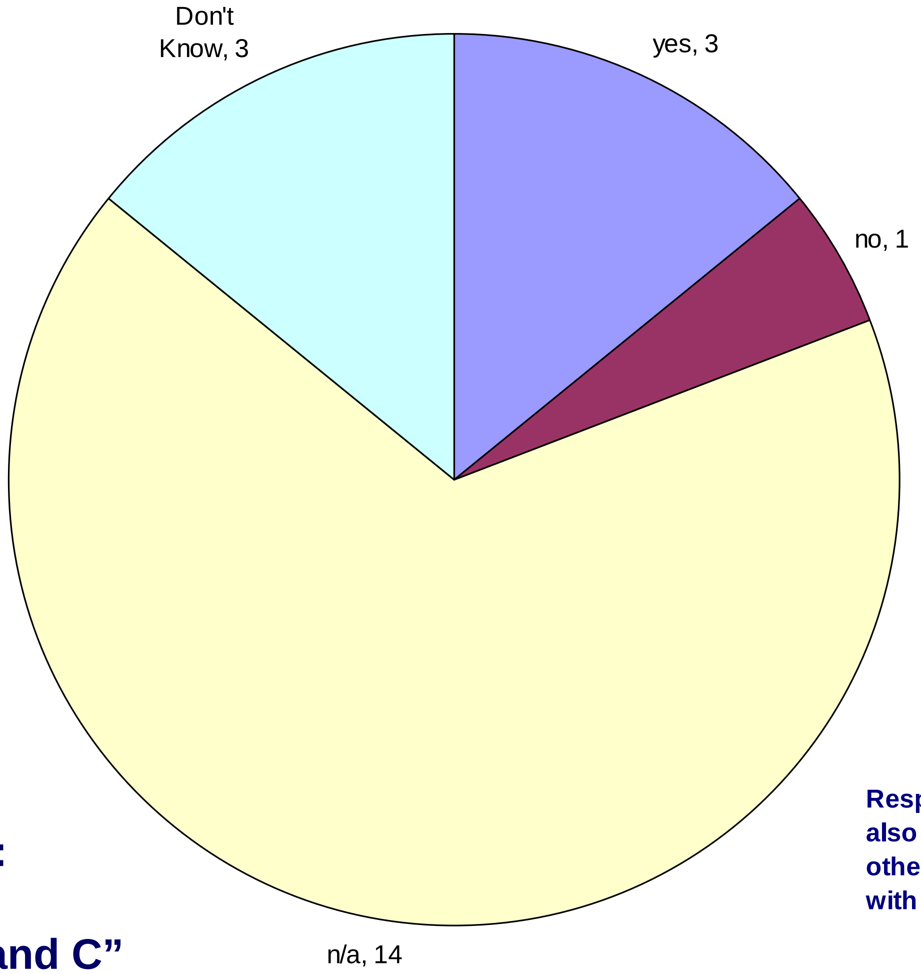


Note there is quite a disconnect between these responses and the number of respondents indicating they were using inheritance with dynamic dispatch...

OOTiA Survey

Survey Responses

For OO work involving inheritance and dynamic dispatch at levels B and C, are you ensuring compiler generated dispatch tables, or their equivalent, are being covered via requirements based tests?



Responses here were also inconsistent with other questions on inheritance with dynamic dispatch

One interesting response:

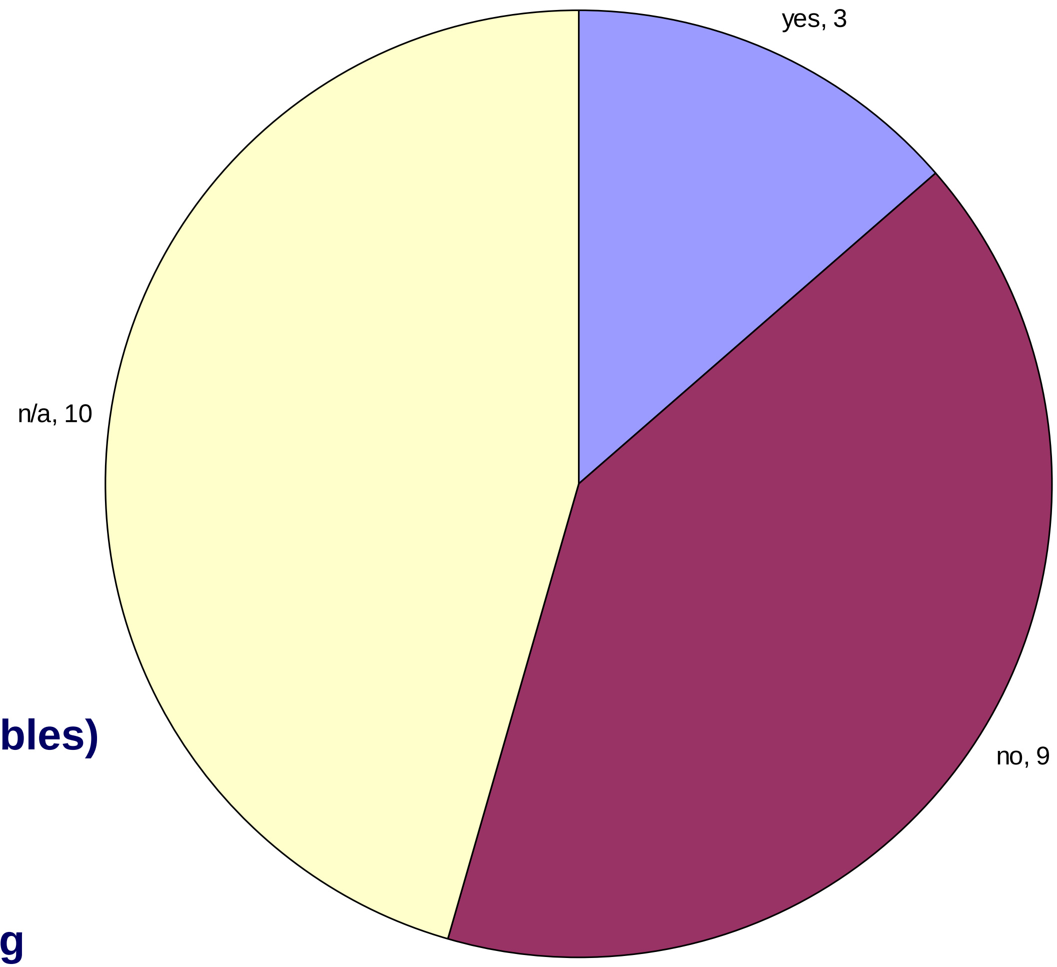
“Not required for level B and C”

OOTiA Survey

Survey Responses

Do you feel you have good tool support for verification issues surrounding inheritance and dynamic dispatch?

If not, what would you like to see made commercially available?



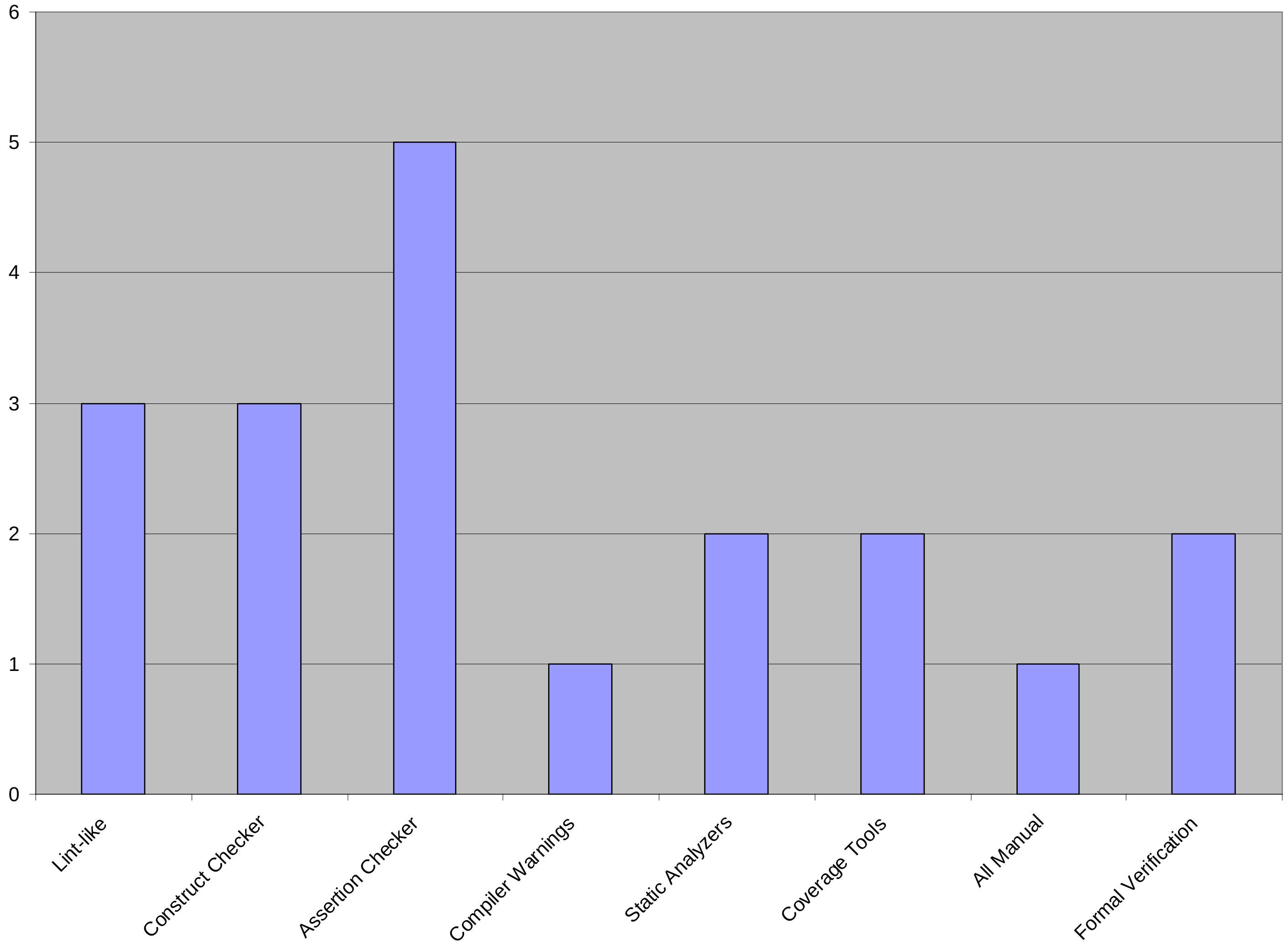
Most cited “needed tools” are tools to ensure compiler supplied code (e.g., dispatch tables) is fully covered

“Yes” responses correlate to sophisticated developers having well-defined language subsets

OOTiA Survey

Survey Responses

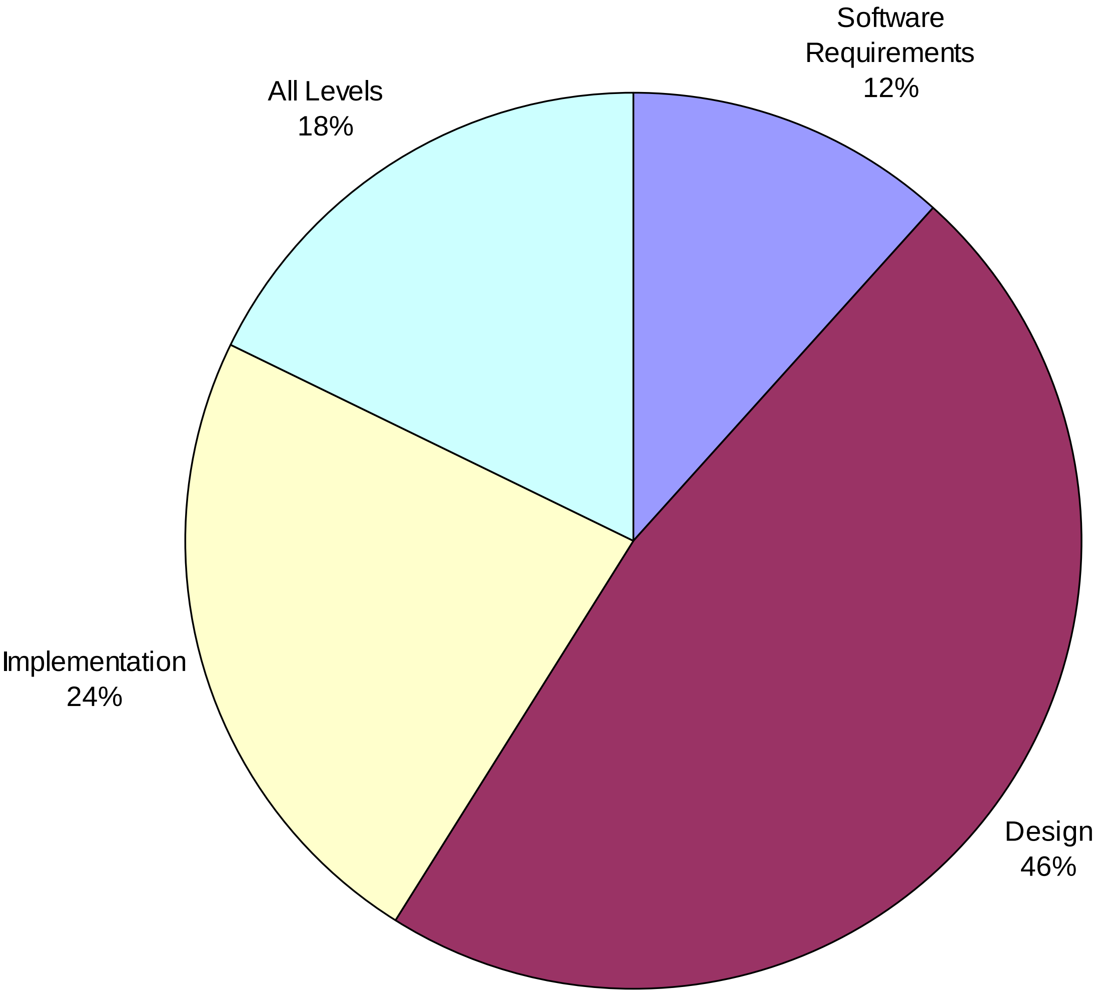
What types of tools do you use with your OO work to reduce manual verification effort (e.g., lint type tools, type and interface checkers, language construct checkers, assertion checkers, and so on)?



OOTiA Survey

Survey Responses

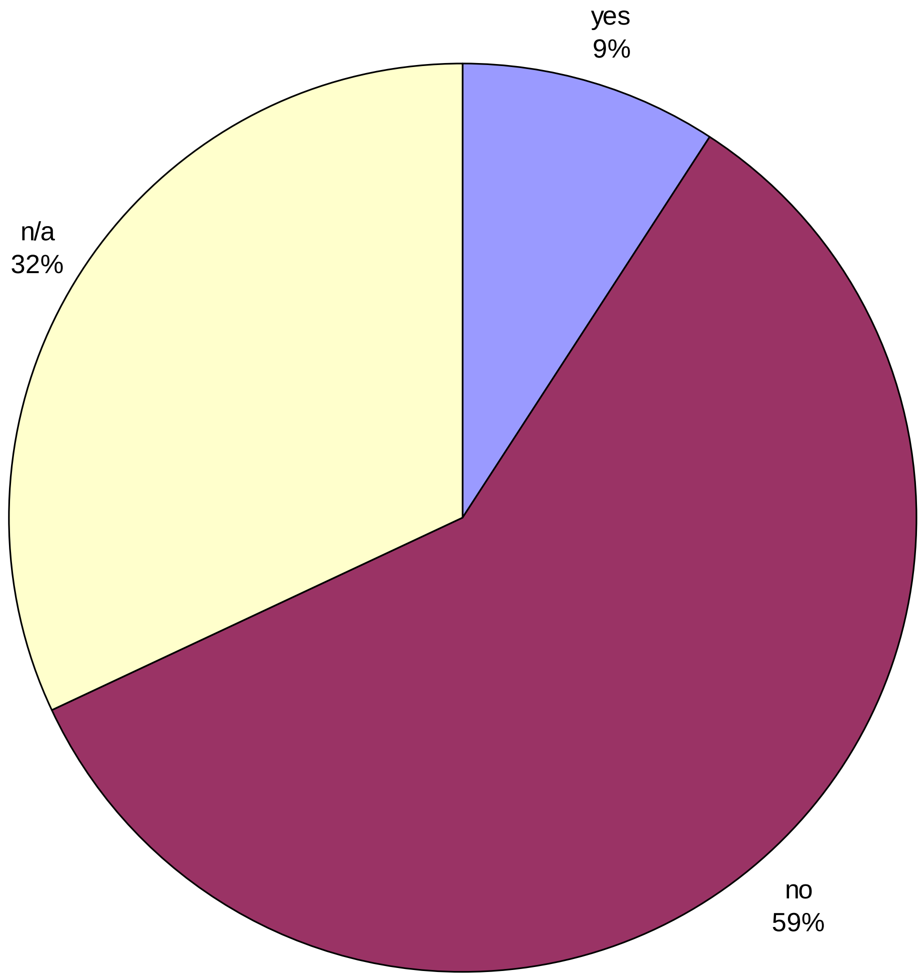
For the portions of your design that are OO, are you using OO techniques from systems analysis through implementation or do you introduce OO techniques somewhere downstream of systems analysis work? If you are willing to share, at what point do you transition to OO techniques?



OOTiA Survey

Survey Responses

Have traceability issues been difficult to deal with, and if so, what would you do differently to reduce traceability issues?

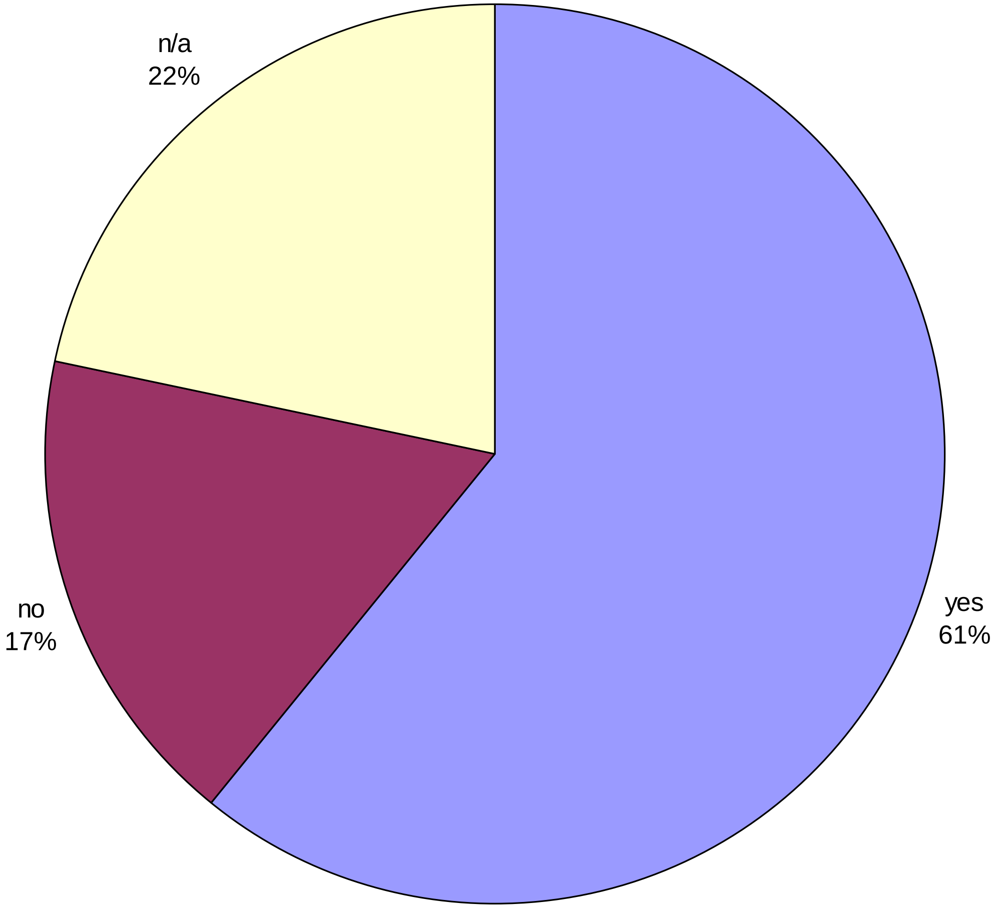


Only one respondent noted traceability challenges from source to compiler generated code (e.g., dispatch tables and subclassing issues) – the remainder only addressed requirements to source issues

OOTiA Survey

Survey Responses

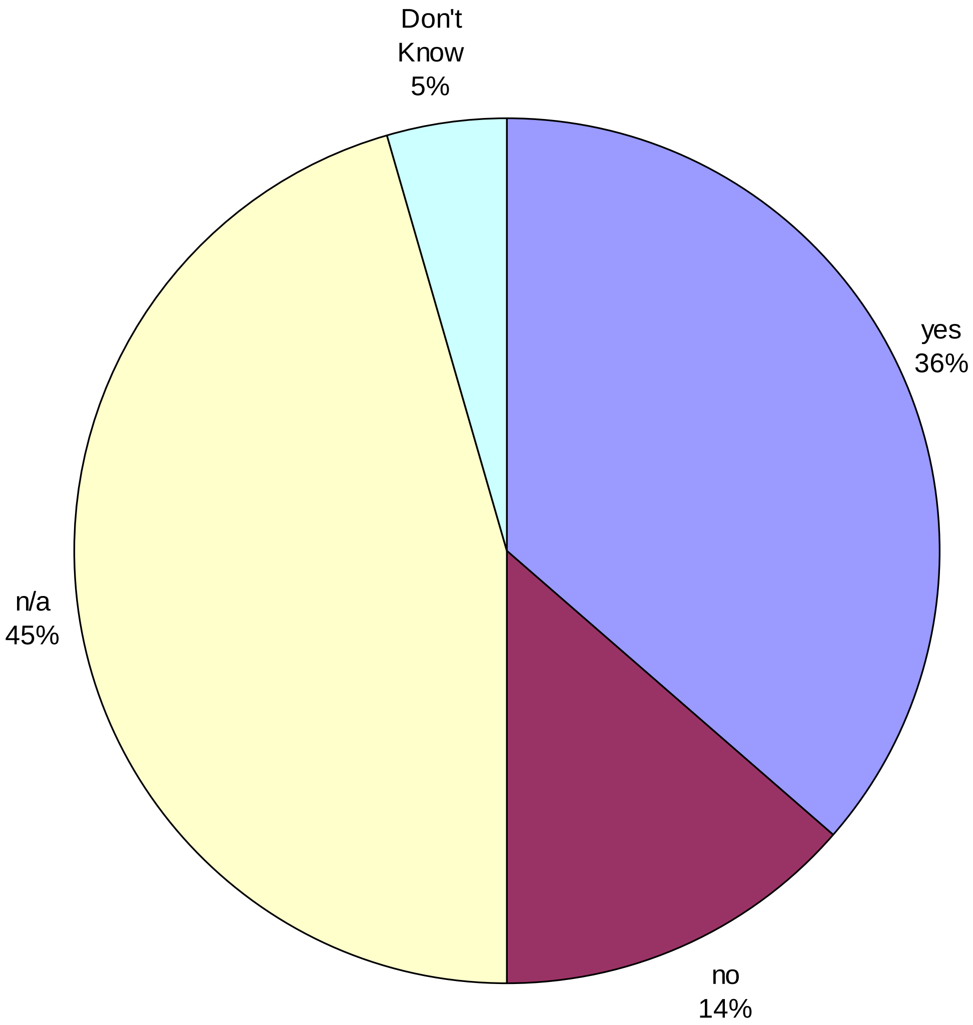
Have you defined a "safe subset" of the computer language you have chosen to use? (In this context, "safe subset" could mean prohibiting the use of constructs or techniques that make OO applications difficult to verify – concepts such as inheritance, polymorphism, nested template classes, dynamic object creation and so on.)



OOTiA Survey

Survey Responses

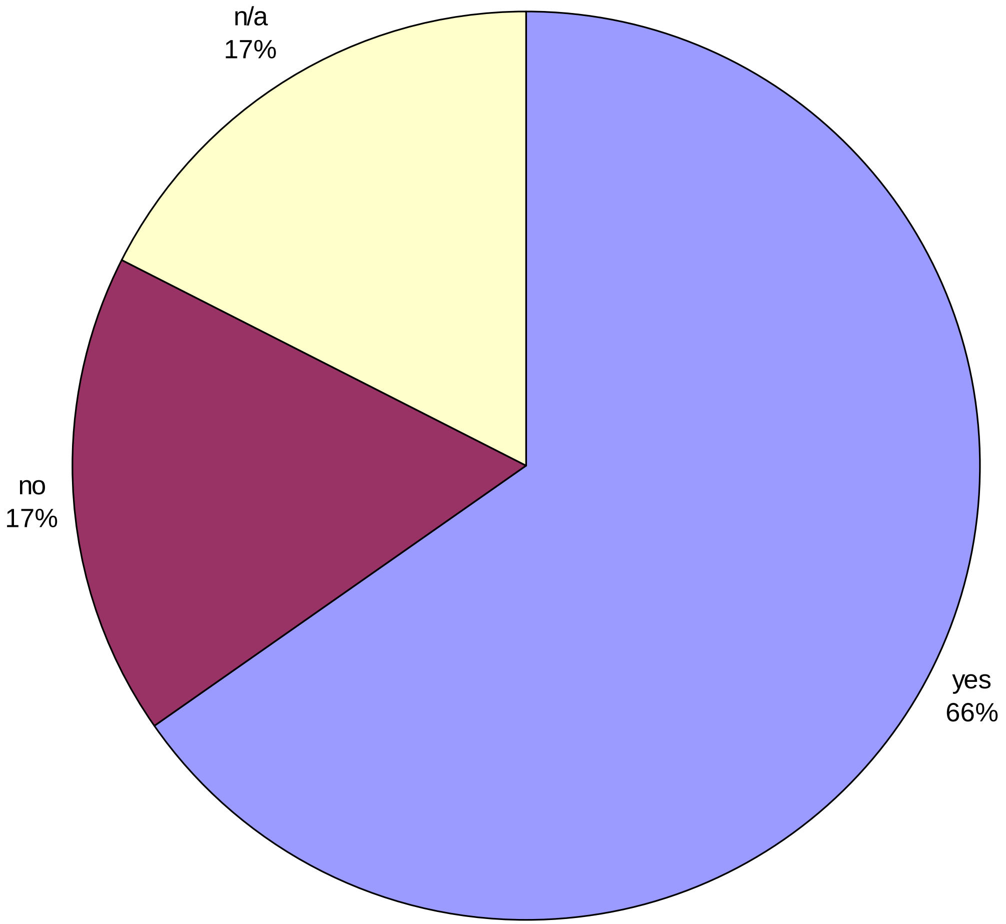
If you develop OO software at multiple criticality levels, do you vary your requirements, design and/or coding standards based on the criticality level?



OOTiA Survey

Survey Responses

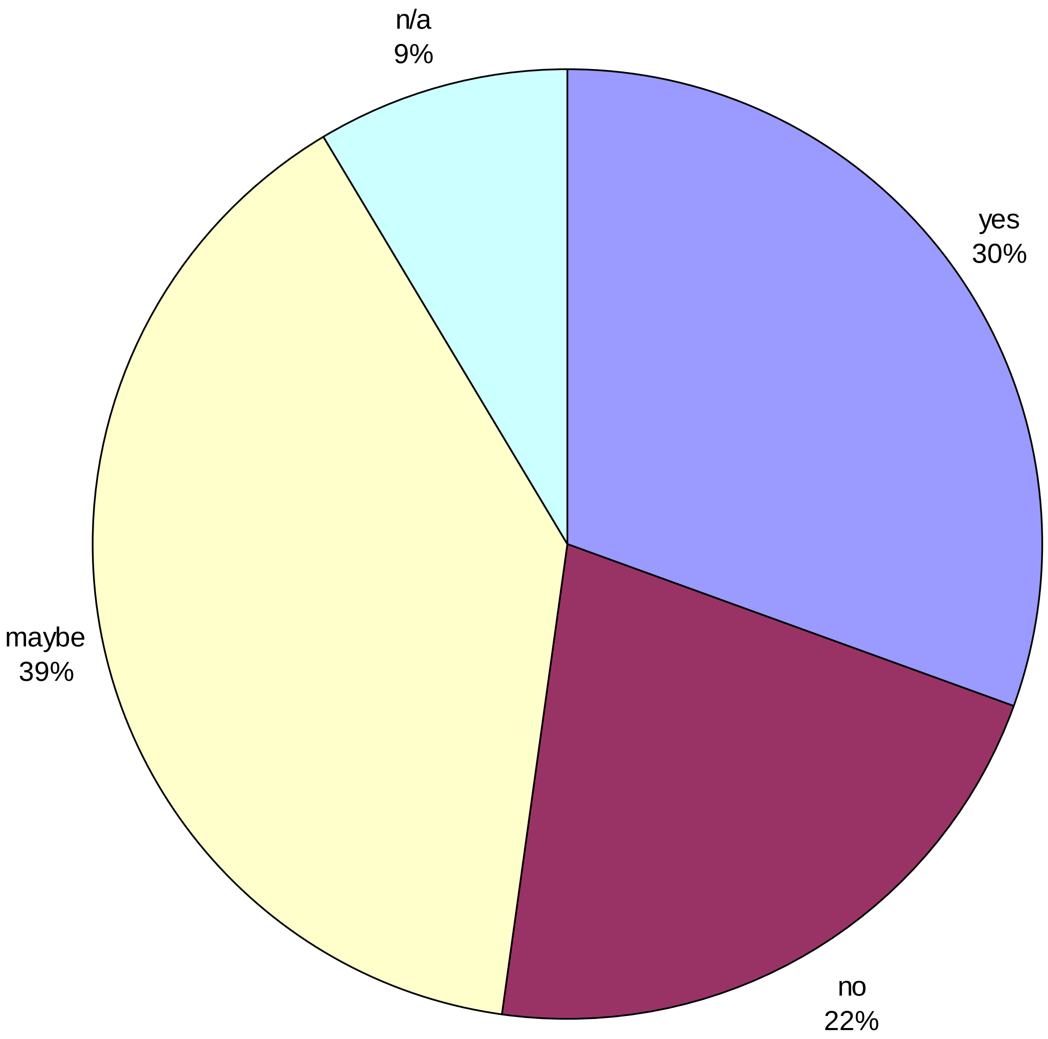
Have you found that you have needed to develop specific standards (requirements, design, code, or even test) that help you control your OO processes?



OOTiA Survey

Survey Responses

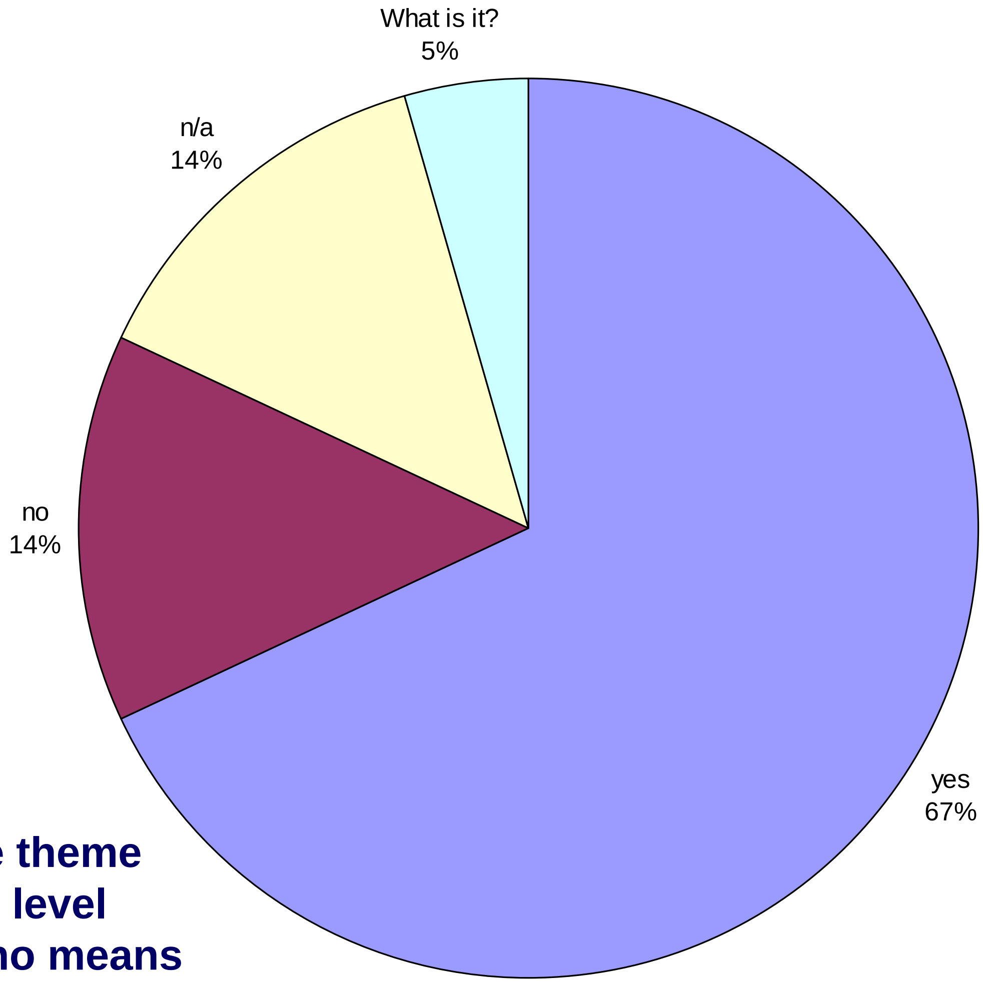
If you started a new program today, would you use OO techniques to see the program to completion or would you stick with a straight procedural approach? Would your answer vary according to the criticality level?



OOTiA Survey

Survey Responses

Has the OOTiA Handbook been useful to you? (If yes, in what ways? If no, what should be examined for improvement?) Note, there is an RTCA/EUROCAE committee/subgroup (SC-205) looking at OO techniques - the results of this survey will be made available to that subgroup.



A common response theme seemed to be “helps level playing field but by no means perfect”

OOTiA Survey

Survey Responses

What additional comments would you like to make about OO techniques in the realm of software development and verification for commercial aviation applications?

- Responses were widely varied – a common theme seemed to be:

  “OO is here to stay – good guidance needs to be developed as soon as possible; SW level modulation and DO-178B-like objectives should also be a part of that guidance”

OOTiA Survey

Possible *Survey Interpretation*

- OO techniques are here to stay but may not be as prevalent at higher software levels as initially thought
- Most applicants are being quite conservative in terms of utilizing inheritance with dynamic dispatch
- OO verification techniques, in some cases, may need further improvement
- Certification authorities appear to be involved a little over half the time – more involvement may be beneficial
- The OOTiA Handbook is useful but could be improved
  - Software level modulation for guidance or guideline material may be helpful
  - Objectives versus, or in conjunction with, a series of training chapters may be desirable
- Tooling support could be improved in the areas of compiler generated functionality – e.g. inheritance, dispatch tables, templates and so on